Proceedings of the APAN – Research Workshop 2017
ISBN 978-4-9905448-7-4

Discriminating DRDoS Packets using Time Interval Analysis

Daiki Noguchi, Tatsuya Mori, Yota Egusa, Kazuya Suzuki, and Shigeki Goto

***Abstract*— The Distributed Reflection Denial of Service (DRDoS) attack represents a critical security threat. As such attacks generate unidirectional traffic, it is difficult for the targets to protect themselves. To mitigate against such attacks, defense mechanisms must be installed on backbone networks, to detect and block the attack traffic before it reaches the final destination. Conventional approaches monitor the traffic volume, and assume that an attack is in progress if the observed volume exceeds a certain threshold. However, this simple approach allows the attacker to evade detection by adjusting the traffic volume. In this study, we proposed a novel approach that accurately detects DRDoS attacks using the time intervals between the arriving packets. We applied a K-means clustering algorithm to identify the appropriate threshold value. The proposed algorithm was implemented at a real data center, and the results demonstrated the high level of accuracy that our approach can achieve.**

***Index Terms*—DDoS, DRDoS, NTP, Time interval, K-means**

I. Introduction

Distributed Denial of Service (DDoS) attacks pose a severe security threat. Distributed Reflection Denial of Service (DRDoS) is a sophisticated form of DDoS that makes use of open servers. There are four prominent types of DRDoS attack, using the protocols CharGen, DNS, NTP, and SSDP [1]. When these User Datagram Protocol (UDP)-based attacks are generated by DDoS as-a-service, they are known as Booters [2].

Several mitigation techniques are available. Li et al. report that cloud services may be used as botnets [3], allowing attackers to expand the scale of the attack at a low cost, and propose a defense called srcTrace. Arbor Networks offers a Peak Flow SP for service providers that monitors huge amounts of backbone traffic to detect malicious packets [4]. These approaches assume that malicious packets will have a large traffic volume. In reality, however, some attacks may not be easy to detect by measuring traffic volumes. For example, the packet size of an HTTP GET request is small, while the response from the Web server is lengthy [5]. The UDP query packets that invoke DRDoS attacks are also small.

In this study, we proposed a novel approach to detecting DRDoS attacks that use small packets. The key is the leverage of time interval analysis. The proposed method compares the time intervals between packets. After removing outliers, it then calculates a threshold value and applies a K-means clustering algorithm to the time intervals.

The rest of this paper is organized as follows: Section II describes the mechanism and introduces the terminology of DDoS and DRDoS attacks. Section III discusses those existing methods that are relevant to the current study. In Section IV, we set out our goals. The proposed method is described in Section V. Section VI reports on an evaluation of the novel method, carried out in a data center. Some further issues are explored in Section VII, and our conclusions are presented in Section VII.

II. DDoS and DRDoS Attacks

In Q4 of 2016, DRDoS attacks occurred more frequently than SYN Flood attacks. Table I shows examples of DRDoS attacks, which are characterized by their large traffic volume. In 2014, NTP amplification attacks attracted attention because of their catastrophic traffic volumes while using *normal* NTP servers, which are open to the Internet.

Figure 1 shows the mechanism of a DRDoS attack. These use UDP packets such as CharGen, DNS, NTP, or SSDP. Such protocols have longer size *responses*, compared with the short

TABLE I
LARGE VOLUME DRDOS ATTACK INCIDENTS.

Date	Protocol	Traffic rate [Gbps]
Mar. 2013	DNS	300
May 2013	DNS	167
Feb. 2014	NTP	400
Aug. 2015	RPC	100

Daiki Noguchi and Shigeki Goto are with the Department of Computer Science and Engineering, Waseda University, Shinjuku, Tokyo 169-8555 Japan, e-mail: (see http://www.goto.info.waseda.ac.jp).

Yota Egusa and Kazuya Suzuki are with SAKURA Internet inc., Grand Front Osaka Tower A 35F, 4-20 Ofukacho, Kita-ku, Osaka-shi, Osaka 530-0011 Japan, e-mail: y-egusa AT sakura.ad.jp (Yota Egusa), ka-suzuki AT sakura.ad.jp (Kazuya Suzuki).

Manuscript received June 25, 2017

queries. The packet size is amplified by the reflector. For example, the *monlist* command of the NTP protocol replies to a query with the communication history for, at most, 600 devices. This offers a convenient tool for attackers who prefer packet-size amplification. Servers that reply to queries from the Internet are called *reflectors*.

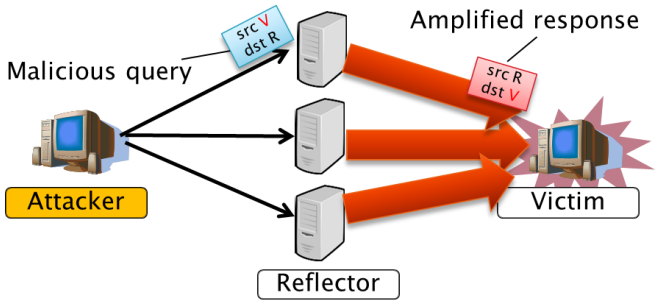

Fig. 1. Mechanism of DRDoS attacks.

The attacker sends malicious queries to these reflectors while spoofing the source IP address as that of a victim. The reflector then unwittingly returns large-scale responses to that address.

A large number of reflectors exist worldwide, most of which are improperly configured or use a default setting. Several organizations attempt to identify vulnerable servers on the Internet, then notify their administrators [10].

III. Related Works

A DRDoS attack works in two ways. A huge number of packets may be generated, occupying the entire bandwidth of certain communication links. This can be detected by measuring the traffic volumes across the links. This detection method is implemented using an IDS (Intrusion Detection System), in which an alarm is triggered if a traffic threshold is exceeded.

A second DRDoS attack type uses a small query packet that is answered by a longer reply packet. If the number of query packets is large, the computational resources of a server, including memory, CPU, or process tables, may be overwhelmed. This form of attack is difficult to detect by measuring traffic volumes.

Our earlier work, reported in [11], analyzed the time intervals between DRDoS packets, and covered the CharGen, DNS, NTP, and SSDP protocols. These time intervals were used to characterize each attack. The current study extended this approach by applying a clustering method to discriminate between DRDoS attacks and normal communications.

Hayashi et al. proposed a time interval analysis-based method for mitigating HTTP GET request flood attacks on backbone networks [12]. Their approach used two threshold parameters, T_{th} and D_{th}. If two packets arrive at the server within a time T_{th}, they are assumed to be successive (see Figure 2). If the series of successive packets has a longer duration than D_{th}, it is treated as suspicious. However, no specific values for T_{th} and D_{th} were given. In this study, a novel method was proposed for determining the values of T_{th} and D_{th}. The approach was then implemented in a data center to detect DRDoS attacks and assess its performance.

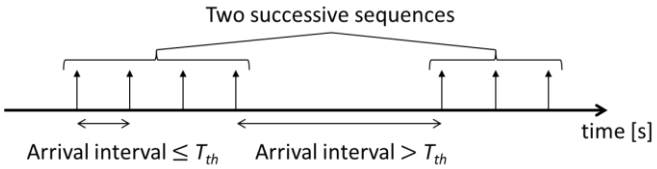

Fig. 2. Packets judged to be successive.

Li et al. [3] used the entropy of a network flow to detect an attacking flow. This assumes that the transmission rate of an attack flow will be larger than that of a legitimate flow. The current study did not assume this. Instead, the analysis was based only on the interval between the arrival times of packets.

IV. Problem to be Solved

A. Environment of a Data Center

The configuration of the data center implementation is shown as Figure 3. This center has its own AS number. The edge router forwards packets between the outside AS and the inside AS.

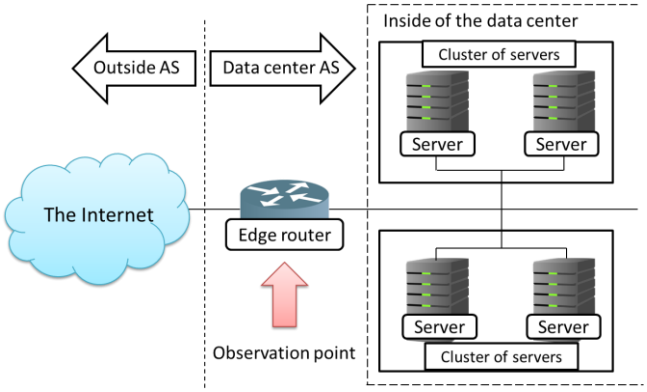

Fig. 3. Configuration of the observation point.

The packets were observed at a certain sampling rate by the edge router. Before data analysis, the source IP and destination IP addresses were anonymized using a hash algorithm, to protect the privacy of the information. Only the headers of the packets were observed. Observations were conducted over the period from November 9, 2016 to December 11, 2016.

The packets were represented in 5-tuple flow format (*src_IP, dst_IP, src_port, dst_port, protocol*). Sampling was conducted randomly, at a rate of ten flows per hour. The key data were the packet lengths and the intervals between packet arrivals. Flows whose UDP source ports were numbered 17, 19, 53, 111, 123, 137, 161, 1900, 3000, and 27960 were picked up, forming the *UDP Port list*. These ports are widely used in DRDoS attacks. We also picked up flows through TCP source port 80, as these are sent by Web servers. Potentially abnormal flows were identified by combining the source port number and the destination port number, as shown in Table II.

TABLE II
DEFINITION OF ABNORMAL FLOWS BY PORT NUMBERS.

Src \ Dst	80	UDP Port List	Over 49151
80		Abnormal	Normal
UDP Port List	Abnormal		Normal

Table III gives a breakdown of the observed flows used in the study. While UDP flows were also observed, these non-NTP packets were rare. Our analysis therefore focused on the NTP flows.

TABLE III
BREAKDOWN OF OBSERVED FLOWS.

	Normal	Abnormal	Total
NTP flow	3,640	14	3,654
HTTP flow	2,311	0	2,311

B. Preliminary Investigation

Since our analysis is based on the time intervals between incoming packets in a flow, the distribution of time intervals is important. Figure 4 shows the time intervals, organized by the range shown in Table IV. Significant differences were observed. Note that the Y-axis of Figure 4 uses a logarithmic scale, and each value differs by at least one order of magnitude from the next. The wide range of time intervals can be seen, and at least five groups could be distinguished. However, it was not clear whether this division by digits was appropriate. We therefore applied a K-means algorithm to the packet intervals to form more robust clusters when *n* equaled five.

TABLE IV
DESCRIPTION OF GROUPS.

Group Number	Conditions
1	$i < 0.1s$
2	$0.1s \leq i < 1s$
3	$1s \leq i < 10s$
4	$10s \leq i < 100s$
5	$100s \leq i$

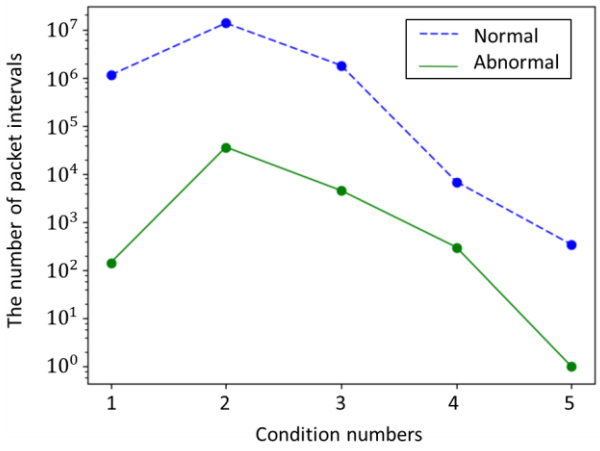

Fig. 4. Packet intervals grouped by five conditions.

C. Identifying a Suspicious Flow

The goal of the study was to propose a new method for discriminating between abnormal and normal flows. The output was therefore labeled either *Suspicious* or *Non-suspicious*. The possible outcomes are shown in Table V.

Our proposed method identifies a flow as suspicious if the time between successive packets in the flow exceeds a certain threshold. We discuss the determination of this value in the following section.

TABLE V
POSSIBLE OUTCOMES.

	Suggest *Suspicious*	Suggest *Non-suspicious*
Abnormal Flow	True Positive (TP)	False Negative (FN)
Normal Flow	False Positive (FP)	True Negative (TN)

The proposed method does not use the *port numbers* to judge whether a sequence of packets is suspicious, because new protocols or port numbers may be used at some future date, allowing attacks to resume. Instead, our method uses only the *time interval between arriving packets*.

V. PROPOSED METHOD

The proposed method has three steps, as shown in Figure 5. In step one, outlier values are removed from each flow. In step two, the packet intervals in each flow are classified. In step three, the threshold value indicating a suspicious flow is derived.

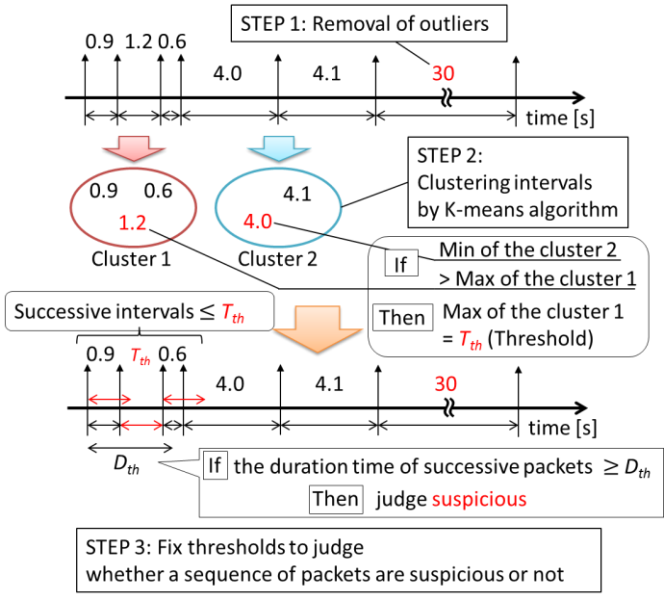

Fig. 5. Flow Chart of the proposed method.

A. STEP 1: Removal of outliers

Outliers may exist in a flow that contains a pause. For example, an attacker may intentionally insert a long pause between certain packets to evade detection. Such outliers would constitute noise when calculating the threshold values

in step two. Any outliers were therefore removed, using the ChangeFinder algorithm [14] shown in Figure 6.

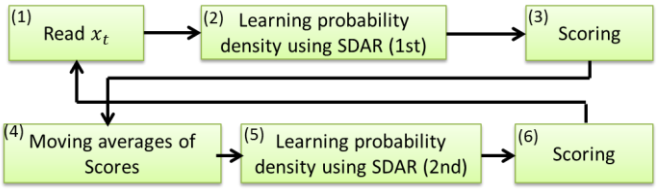

Fig. 6. Two-phase learning process of the ChangeFinder algorithm.

ChangeFinder applies two phase learning. First, input data are analyzed using the Autoregressive (AR) model, then the sequentially discounting AR model (SDAR) learning is applied. The AR model is given by Equation (1) where y_t is

$$y_t = \sum_{i=1}^{o} a_i y_{t-i} + w \qquad (1)$$

the data on the packet intervals, a_i is the AR coefficient, o is the order, and w is white noise, which follows a distribution whose average is zero. ChangeFinder learns the probability density P_t, of the packet intervals x_t by applying the SDAR algorithm, then calculates the outlier scores $m(x_t)$ using Equation (2).

$$m(x_t) = -\log p_{t-1}(x_t|x^{t-1}) \qquad (2)$$

Here, $p_{t-1}(x_t|x^{t-1})$ is the conditional density function of x_t against the stochastic process p, *and* x^{t-1} is the series (x_1, x_2, … x_{t-1}). The scores indicate the degree of separation between the values predicted by the AR model and x_t. The AR model mainly assumes stationary data, so an important feature of ChangeFinder is its ability to handle non-stationary data by applying the SDAR algorithm. Figure 7 shows how ChangeFinder detects outliers in the data. These outlier scores are normalized regardless of the data range. In this study, we set a threshold value of 50 for the detection of outliers.

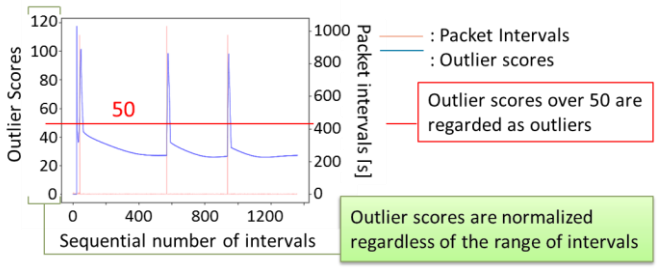

Fig. 7. Packet intervals and outlier scores.

B. *STEP 2: Clustering by K-means*

The K-means clustering algorithm [16] was used to discriminate between successive packet sequences from the terminated packet sequences. The two threshold values, T_{th} and D_{th}, were determined in Step 3. The K-means algorithm attempts to minimize the value of Equation (3):

$$\sum_{i=1}^{n} \sum_{x \in C_i} ||x - \mu_i||^2 \qquad (3)$$

Here, C_i $(1 \le i \le n)$ represents a cluster, n is the number of clusters, x is an element in the cluster, and μ_i is the centroid of the cluster. The K-means algorithm selects an appropriate μ_i for each cluster, in order to minimize the value of Equation (3).

We used the K-means algorithm in the scikit-learn package [18], and applied it to each normal and abnormal flow. Each cluster was ranked in ascending order, based on its maximum interval. The values Q, R, and U in Table VI were then calculated for each cluster, and the value of U used to compare the clusters. When calculating Q, we rounded off the packet interval time to two decimal places.

TABLE VI
DESCRIPTION OF VALUES FOR DETECTION.

Values	Description
Q	Maximum value of the number of duplicated packet intervals in the cluster
R	Total number of packet intervals in the cluster
U	Q × R

The cluster whose U value (=Q×R) is largest is likely to include a large number of intervals between successive packets. Li [3] noted an apparent difference between short packet intervals and long ones. Our observations confirmed this.

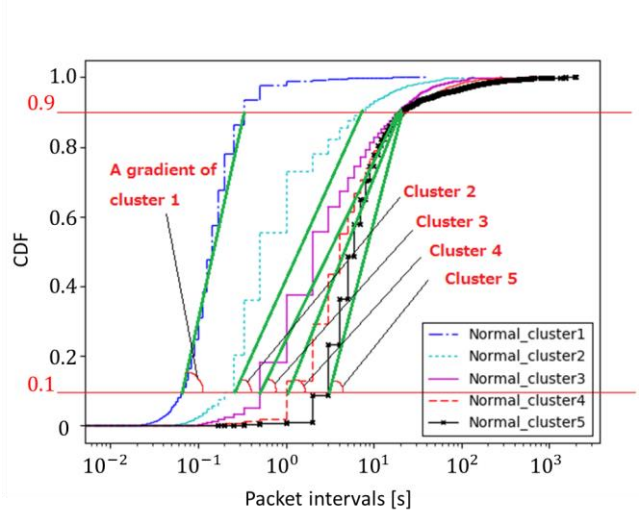

Fig. 8. Example distribution of packet intervals in different clusters.

Within a flow there are five clusters, each of which is a set of packet intervals. Figure 8 shows a distribution graph (CDF) of the packet intervals, in which gradient lines connect the points at CDF values of 0.1 and 0.9. In the example, there are five distribution graphs of packet intervals for each cluster, with different gradients. Combining all the flows and plotting the CDFs and gradients produced Figure 9, in which the distribution and gradient of merged clusters 2, 3, 4, and 5 are also shown.

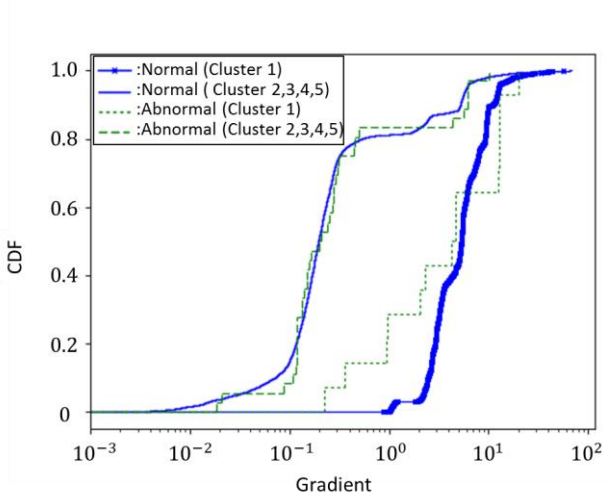

Fig. 9. Distribution of gradients of all flows.

The gradient line is steep if the distribution of packet intervals is concentrated. As shown in Figure 9, the CDF of cluster 1 for both normal and suspicious flows was larger than the CDF of the merged clusters 2, 3, 4, and 5. This suggested that shorter packet intervals were concentrated. However, longer packet intervals were also present. Cluster 1 was therefore assumed to include a large number of packet intervals. However, as it was not clear that the same result could be seen in each flow, the value Q was used.

Because short intervals were more numerous than large intervals, we adopted the value R, as can be seen from Figure 4. This showed that the majority of intervals were small.

As noted above, successive intervals should be short. We named the cluster whose U was largest C_{succ}, and the cluster next to C_{succ}, and whose maximum interval was larger, C_d. This is shown in Figure 10. Most of the packet intervals in C_d were not successive.

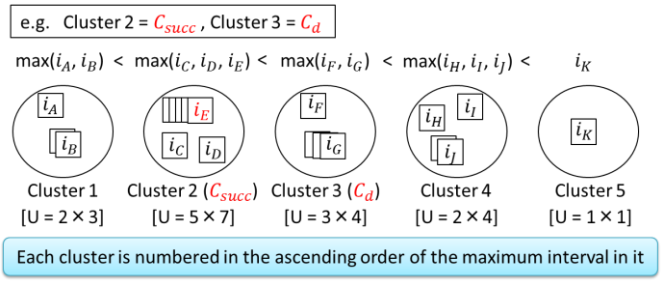

Fig. 10. Examples of clusters and packet intervals.

In Figure 10, interval i_E is within C_{succ}. Because C_{succ} has the largest U, i_E is a successive interval, whereas i_F is within C_d, and it is non-successive.

C. STEP 3: Determining the values

To detect suspicious flows, the optimal threshold D_{th} must be found.

1) Longest interval time T_{th} for successive packets

We set T_{th} to the maximum packet interval time in cluster C_{succ}. This threshold value was used in (2).

2) Formation of successive packet sequences

For each flow, we collected all the interval times from cluster 1 to C_{succ}. For each flow, we formed a successive sequence of packets that fell within the maximum interval time T_{th} in (1). We then calculated the total duration of a sequence as the sum of the successive arrival times. For a normal flow, we used all successive sequences. For an abnormal flow, we used the maximum successive arrival time.

3) Fix D_{th}

Finally, we calculated the distribution of successive duration times of the normal and abnormal flows, represented in the CDFs. The optimal time D_{th} is given when Equation (4) takes the maximum value.

$$CDF(normal\ arrival\ time) - CDF(abnormal\ arrival\ time) \quad (4)$$

The proposed method uses the derived value of D_{th} as the threshold for detecting a suspicious flow. If the time between successive packets equals or exceeds this threshold, an abnormal flow is suspected.

VI. Evaluation

A. Threshold Value: D_{th}

To evaluate our method, the proposed method was applied to the equipment in a data center, as shown in Figure 3. The algorithm produced a threshold value of D_{th} = 4.0 sec.

B. Performance of the proposed method

Figure 11 shows the results. These follow the ground truth of flows defined by port numbers from Table II, in which an abnormal flow has the source port number 80 (HTTP) and a destination port number 123 (NTP) or a source port number 123 (NTP) and the destination port number 80 (HTTP). Table VII shows the outcomes, where S is the time between the arrival of successive packets in the real traffic. The True Positive rate (TP) was 100%, and the False Positive rate (FP), while not zero, was low at 5%. For the analyzed packets, most flows were normal. If a high (FP) were found, extra resources would be needed to investigate those suspicious flows that turn out to be normal.

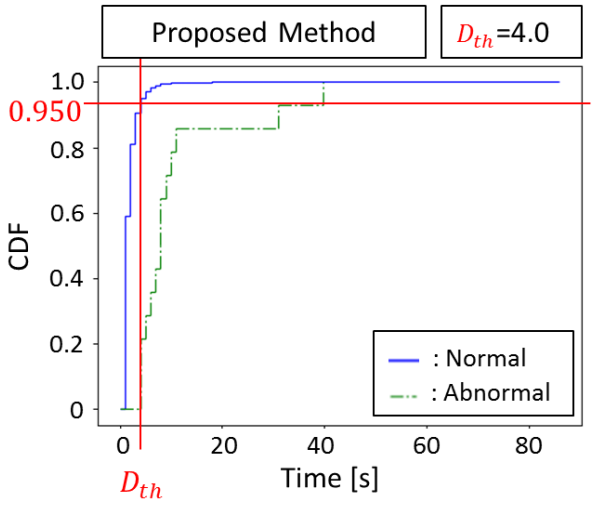

Fig. 11. Distribution of duration times.

TABLE VII
RESULTS OF EVALUATION.

	$S \geq 4.0$	$S < 4.0$
Abnormal Flow	100.0%	0.0%
Normal Flow	5.0%	95.0%

VII. DISCUSSION

A. *Effect of outlier removal*

When the outlier values were not removed, different results were produced. Using the same threshold value of D_{th} = 4.0, the FP rate rose to 14.2%. Figure 12 shows the results without removal of the outliers. It can be seen that the performance was inferior to that reported in Figure 12, demonstrating the effectiveness of outlier removal.

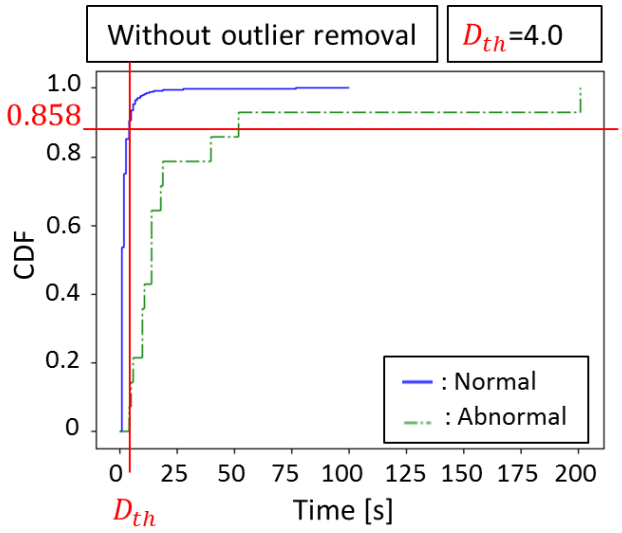

Fig. 12. Distribution of interval times.

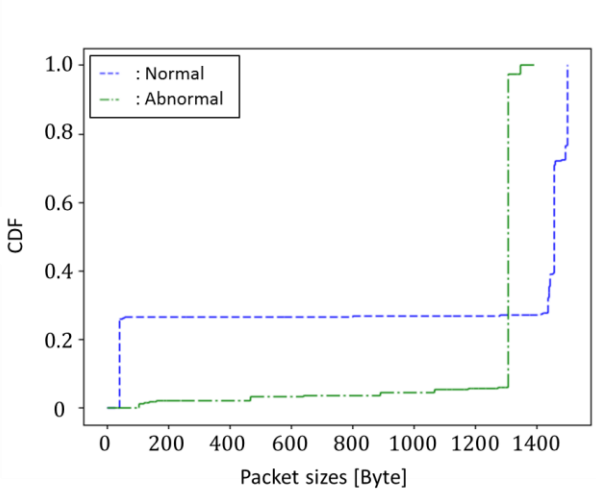

Fig. 13. Distribution of packet sizes across all sampled flows.

B. *Observed packet sizes*

In the Introduction, we noted that the UDP query packets used to invoke DRDoS attacks are small. Figure 13 shows the distribution of packet sizes across all sampled flows in the data center. This demonstrates that there exists no threshold value that allows normal HTTP packets to be distinguished from NTP attacks based on packet size alone.

VIII. CONCLUSIONS

This study proposed a practical method for detecting DRDoS attacks by analyzing the time intervals between the arrival timestamps of packets. Threshold values were determined after outlier removal, using a K-means clustering algorithm.

An evaluation experiment demonstrated that our proposed method is superior to the conventional DRDoS detection method, based on measurement of traffic volume.

This study addressed only the NTP protocol. In future work, we will investigate DRDoS attacks using other protocols, including CharGen, DNS, and SSDP.

ACKNOWLEDGEMENTS

A part of this work was supported by JSPS Grant-in-Aid for Scientific Research B, Grant Number JP16H02832.

REFERENCES

[1] Akamai, "Q4 2016 State of the Internet Security Report," https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/q4-2016-state-of-the-internet-security-report.pdf, referred Nov. 5, 2016.

[2] Santanna, José Jair, et al., "Booters—An analysis of DDoS-as-a-service attacks," Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on. IEEE, 2015.

[3] B. Li, W. Niu, K. Xu, C. Zhang, P. Zhang, "You can't hide: a novel methodology to defend DDoS attack based on BotCloud," Applications and Techniques in Information Security, Communications in Computer and Information Science, Springer Berlin Heidelberg, pp. 203 – 214, 2015.

[4] Arbor Networks, "Arbor Networks SP," http://www.arbornetworks.com/, referred Oct. 23, 2016.

[5] IMPERVA INCAPSULA, "HTTP FLOOD," https://www.incapsula.com/ddos/attack-glossary/http-flood.html, referred Oct. 20, 2016.

[6] The Register, "BIGGEST DDoS ATTACK IN HISTORY hammers Spamhaus," https://www.theregister.co.uk/2013/03/27/spamhaus_ddos_megaflood/, referred Oct. 12, 2016.

[7] Internet Initiative Japan, "Problems about DNS Open Resolver," https://www.iij.ad.jp/company/development/report/iir/pdf/iir_vol21_internet.pdf, referred Oct. 12, 2016.

[8] JANOG, "NTP Reflection DDoS Attack Explanatory Document," https://www.janog.gr.jp/wg/doc/ntp-wg-en.pdf, referred Oct. 12, 2016.

[9] Akamai, "Akamai Warns Of 3 New Reflection DDoS Attack Vectors," https://www.akamai.com/jp/ja/about/news/press/2015-press/akamai-warns-of-3-new-reflection-ddos-attack-vectors.jsp, referred Oct. 12, 2016.

[10] Shadowserver, https://www.shadowserver.org/wiki/, referred Jan. 24, 2017.

[11] Daiki Noguchi and Shigeki Goto, Defense against DRDoS Attacks by OpenFlow Switches, Proceedings of the computer security symposium 2016, pp.1183 – 1190, October, 2016. (in Japanese)

[12] Yuhei Hayashi et al., “Evaluation of the attack detection method based on duration of continuous packet arrival,” IEICE technical report 115(488), pp. 53 - 58, 2016. (in Japanese).
[13] Arbor Networks, “Worldwide Infrastructure Security Report Volume X,” http://pages.arbornetworks.com/rs/arbor/images/WISR2014_EN2014.pdf, referred Nov. 15, 2016.
[14] J. Takeuchi, K. Yamanishi, “A unifying framework for detecting outliers and change points from time series,” IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 4, pp. 482 – 492, 2016.
[15] NTT DATA Mathematical Systems Inc., “Change point detection, ChangeFinder,” http://cl-www.msi.co.jp/reports/changefinder.html, referred Oct. 20, 2016.
[16] MacQueen, J. B.,“Some Methods for classification and Analysis of Multivariate Observations,” Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability 1, University of California Press, pp. 281 - 297, 1967.
[17] Python Software Foundation, “PyPI – the Python Package Index,” https://pypi.python.org/pypi, referred Oct. 21, 2016.
[18] scikit learn, “scikit-learn,” http://scikit-learn.org/stable/index.html, referred Oct. 24, 2016.
[19] IMPERVA INCAPSULA, “SNMP REFLECTION / AMPLIFICATION,” https://www.incapsula.com/ddos/attack-glossary/snmp-reflection.html, referred Oct. 22, 2016.
[20] Alejandro Nolla, “Amplification DDoS Attack With Quake3 Servers,” http://blog.alejandronolla.com/2013/06/24/amplification-ddos-attack-with-quake3-servers-an-analysis-1-slash-2/, referred Oct. 22, 2016.

Daiki Noguchi Daiki Noguchi received the B.S. degree in Computer Science and Engineering from Waseda University in March, 2016. He is now a master student at Department of Computer Science and Communications Engineering, Waseda University. His research interest covers Future Internet and Cyber Security.

Tatsuya Mori Tatsuya Mori is currently an associate professor at Waseda University, Tokyo, Japan. He received B.E. and M.E. degrees in applied physics, and Ph.D. degree in information science from the Waseda University, in 1997, 1999 and 2005, respectively. He joined NTT lab in 1999. Since then, he has been engaged in the research of Internet measurement and security. He is a member of ACM, IEEE, IEICE, IPSJ, and USENIX.

Yota Egusa Yota Egusa received the B.S. degree in Computer Science and Engineering from Osaka University in March, 2014. He is now a technology executive officer of SAKURA Internet inc.

Kazuya Suzuki Kazuya Suzuki received Associate Degree of Engineering from National Institute of Technology Ibaraki College in 2014. He is now a network engineer of SAKURA Internet inc.

Shigeki Goto Shigeki Goto is a professor at Depart-ment of Computer Science and Engineering, Waseda University, Japan. He received his B.S. and M.S. in Mathematics from the University of Tokyo. Prior to becoming a professor at Waseda University, he has worked for NTT for many years. He also earned a Ph.D in Information Engineering from the University of Tokyo. He is the president of JPNIC. He is a member of ACM and IEEE, and he was a trustee of Internet Society from 1994 to 1997.